IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of: Serial No. 10/021,016 Application of: Alexandre Drobychev, James Kong, Nirupama Mallavarupu, Ching-Wen Chu Filed: December 19, 2001 For: METHOD AND SYSTEM FOR THE DEVELOPMENT OF COMMERCE SOFTWARE APPLICATIONS	Confirmation No.: 8774 Art Unit: 2191 Examiner: Vo, Ted T. Customer No.: **32658**

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPELLANT'S BRIEF UNDER 37 CFR 41.37 - AMENDED

I. Real Party in Interest

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
USA

II. Related Appeals and Interferences

No other appeals or interferences are currently known to Appellant that will directly affect, be directly affected by, or have a bearing on the decision to be rendered by the Board of Patent Appeals and Interferences in the present appeal.

III. Status of Claims

Claims 1-16, 18-26, and 29-32 are pending in the application, with claims 17, 27, and 28 being cancelled. No claims have been allowed, and all pending claims stand rejected.

Claims 1-16, 18-26, and 29-32 stand rejected under 35 U.S.C. §102(b) as being anticipated by Netscape Application Builder, User Guide, 1999 ("NAB").

The rejection of independent claims 1, 10 and 21 as being anticipated by NAB is the subject of this appeal.

IV. Status of Amendments

Appellant believes that all claim amendments have been entered.

Claims 1-16, 18-26, and 29-32, including any proposed and entered claim amendments, are provided in the attached Claims Appendix.

V. Summary of Claimed Subject Matter

The rejection of independent claims 1, 10, and 21 are at issue in this Appeal. The following concise explanation of the subject matter defined in each independent claim involved in this Appeal, claims 1, 10, and 21, refers to the specification as originally filed by page and line numbers, and to the drawings by reference characters. In addition, a concise explanation of the subject matter defined in dependent claims 2-9, 11-16, 18-20, 22-26, and 29-32 is also provided. It is noted that the rejection of dependent claims 2-9, 11-16, 18-20, 22-26, and 29-32 are not at issue and not argued separately.

Claim 1 states:

1. A computer system run-time platform for providing features and services for commerce software applications, and operatively adaptable to any server platform capable of server-side presentation logic, the applications platform comprising:

a software portion configured to provide access to, and caching of, data elements, including a data and object repository, independent of the run-time platform for providing commerce software applications;

a software portion configured to inherit hierarchical application logic from the commerce applications platform;

a software portion configured to provide static and dynamic presentation data for presentation by any server capable of server-side presentation logic;

a software portion configured to maintain permanent and session application data persistent across user request boundaries during a single user session; and

a software portion configured to enable access to a business object during the user session.

Figure 3 depicts the commerce application platform layer 230 and the application layer 240. As is described in paragraph [22] beginning on the bottom of page 7, line 21, the commerce application platform layer 230 includes various features and services. These features and services include a user authentication feature 302, a rule engine 304, business objects 306, an application logic interface 307, a presentation logic interface 308, a persistent object framework 310, permanent and session data management 312 and a data store interface 314.

Claim 1 describes a run-time application platform for providing commerce software applications having a set of features and services designed to facilitate the development and use of business applications useable on any platform that supports standard server-side presentation logic. This run-time platform is configured to provide access to, and caching of, data elements, including a data and object repository independent of the run-time platform providing the commerce software applications. A description of the caching of data elements can be found in the text beginning on line 9 of page 19, paragraph [74] and continuing through line 7 of page 20, paragraph [75]. The data store interface 314 (paragraph [23], page 8, line 6) provides access to the data stores or elements of the present invention including data and object repositories. As explained on page 12, line 1 through 11, a data element or data item is a category that can be input or output by an application. These are referred to in the text as a DOR entry. A DOR entry refers to the data or object repository. This aspect of the claim can occur independent of the run-time platform as described on page 8, lines 12-14 (paragraph [24]).

Claim 1 also describes inheriting hierarchical application logic from the commerce applications platform and providing static and dynamic presentation data for presentation by any server capable of server-side presentation logic. The application logic interface 307 (paragraph [26], page 9, line 1) provides applications with the ability to access (inherit) platform features and services using servlets. Servlets, also described in paragraph [27], gain and validate input data as well as perform business functions on the data to compose pages or frames to be presented to the user.

Static and dynamic presentation data for presentation by any server capable of server-side presentation logic is explained at lines 9 through 22 of page 6 (paragraphs [18-19]). The presentation logic interface (paragraph [28], page 9, line 15) provides the applications with access to screen templates. These templates enable the application to combine HTML and functional aspects to pages presented to the clients. Tags or placeholders within the HTML codes create the functional aspects for the template.

Scriptlets are then used to dynamically generate data for presentation through the Java servlets which is applicable to any server capable of server-side presentation logic. The text found on pages 10-14 and paragraphs [32] through [60] provides specific detail and examples of the aforementioned process.

The invention of claim 1 is further configured to maintain permanent and session application data persistent across user request boundaries during a single user session and is also configured to enable access to a business object during that user session. Permanent and session data management 312 is discussed in paragraph [62] found on page 14, beginning on line 16. Both types of data persist across user request boundaries. Permanent data resides in a Name, Type, Value ("NTV") tree and is available in subsequent servlets of the same user session. The four session data methods are found described in paragraph [64] beginning on line 14 of page 15.

Business objects 306 and Persistent Object Framework support 310 (paragraph [69], page 17, line 11) provide management of objects and business objects for use by the application logic interface 307. Through the Persistent Object Framework, persistent objects, including business objects, can be created and used with the various business applications.

Access to a business objects during the user session can be found described throughout the specification and specifically in paragraph [30] on page 10, lines 7-14 and in paragraph [84] on page 22 lines 15-17.

Claim 10 states:

10. A method for implementing a first software application resident on a commerce application platform wherein the commerce application's platform is a run-time platform configured to provide access to data elements, hierarchical inheritance of the first software application logic,

static and dynamic presentation data, access to business objects, and access to permanent and session application data during a single user session, the method comprising:

providing presentation information by the first software application seeking input data from a user;

receiving input data from a data repository for use by the first software application, wherein the data repository is independent of the commerce application platform;

passing the input data to the commerce application platform for validation;

validating the data by the commerce application platform;

providing, by the commerce application platform, business object functionality to the application;

preparing presentation information by the application based upon the business object functionality for presentation by any server capable of server-side presentation logic; and

accessing the permanent and session application data persistent across user request boundaries during the single user session.

Claim 10 describes a method for implementing a software application resident on a commerce application platform wherein the commerce application's platform is a run-time platform configured to provide access to data elements, hierarchical inheritance of the software application logic, static and dynamic presentation data, access to business objects, and access to permanent and session application data during a single user session. The method comprises the steps of providing presentation information by first seeking input data from a user. Input data is thereafter received from a data repository, the data repository being independent of the commerce application platform, for use by the software application. The input data is then passed to the commerce application platform for validation.

Business object functionality to the application is provided by the commerce application platform. The business object functionality is in turn the basis for the application's preparation of presentation information for presentation by any server capable of server-side presentation logic. Finally, the method concludes by accessing the permanent and session application data persistent across user request boundaries during the single user session.

In addition to the citations presented above with respect to claim 1, the method of the present invention is also depicted in Figure 4 and described in the text beginning at paragraph [81] on page 21, line 16. As described in paragraph [82] beginning on line 1 of page 22, the process includes receiving 410 data input by a user for use by a software application. The data store interface 314 (paragraph [23], page 8, line 6) provides access to data stores or elements including data and object repositories that are then used in the software application. As explained in paragraph [36] on page 12, line 1 through 11, a data element or data item is a category that can be input or output by an application. These are referred to in the text as a DOR entry. A DOR entry refers to the data or object repository.

The input data is then passed 420 to the commerce application platform for validation. The received data is validated (paragraph [83], page 22, line 8) to ensure that appropriate security measures and other actions can be taken with the data. The validated data is then used (paragraph [84], page 22, line 14) to determine business objection functionality 440. Data stores, including permanent or session data, may also be accessed through the business objects to provide information needed to perform the current business object functionality.

Presentation information is generated 450 (paragraph [85], page 23, line 1) based upon the business object functions. The presentation logic interface (paragraph [28], page 9, line 15) provides the applications with access to screen templates. These templates enable the application to combine HTML and functional aspects to pages

presented to the clients. Tags or placeholders within the HTML codes create the functional aspects for the template. Scriptlets are then used to dynamically generate data for presentation through the Java servlets which is applicable to any server capable of server-side presentation logic. See text found on pages 10-14 and paragraphs [32] through [60].

Permanent and session application data that is persistent across user request boundaries is accessed during the single user session . Permanent and session data management 312 is discussed in paragraph [62] found on page 14, beginning on line 16. Both types of data persist across user request boundaries. Permanent data resides in a NTV tree and is available in subsequent servlets of the same user session. The four session data methods are found described in paragraph [64] beginning on line 14 of page 15. Access to the data is also discussed on page 15 lines 1-13 in paragraphs [62 - 63].

Claim 21 states:

21. A method for providing services to a first software application residing on a commerce application platform wherein the commerce applications platform is a run-time platform configured to provide access to data elements, hierarchical inheritance of the first software application logic, static and dynamic presentation data for presentation by any server capable of server-side presentation logic, access to business objects, and access permanent and session application data during a single user session, the method comprising:

receiving from the application input data for validation from a data repository, wherein the data repository is independent of the commerce application platform;

validating the input data;

providing business object functionality to the application; and

accessing the permanent and session application data persistent across user request boundaries during the single user session.

Claim 21 also describes a method for providing services to a first software application residing on a commerce application platform. The commerce application platform is a run-time platform configured to provide access to data elements, hierarchical inheritance of the first software application logic, and static and dynamic presentation data for presentation by any server capable of server-side presentation logic.

The method of claim 21 receives input data from the application for validation from a data repository that is independent of the commerce application platform, validates the input data and provides business object functionality to the application. As shown in Figure 4 and described in the specification, the received data is validated (paragraph [83], page 22, line 8) to ensure that appropriate security measures and other actions can be taken with the data. The validated data is then used (paragraph [84], page 22, line 14) to determine business objection functionality 440 which is provided to the application. Data stores, including permanent or session data, are accessed through the business objects to provide information needed to perform the current business object functionality.

While not at issue in this appeal, the limitations found in dependent claims 2-9, 11-16, 18-20, 22-26 and 29-32 are also generally found throughout the specification. Specifically, the present invention includes a data store interface 314 providing access to the data stores of the invention (paragraph [23], page 8, line 6). These stores are formatted in a NTV format. In addition to the aspects of the present invention described above, a rule engine 304 (paragraph [75], page 20, line 1) of the business application platform 230 evaluates attributes associated with a user or a specific transaction. The caching of business objects and non-business Persistent Object Framework objects (paragraph [74], page 19, line 9) makes them easily accessible by any subsequent servlets running in the stateful execution mode.

The specific coding examples found in paragraphs [32-71] (bottom of page 10 to the top of page 18) may assist in a better understanding of the general applicability of the claimed process of the present invention. The Appellant refers the Board to the previous discussion and cited portions of the specification for details regarding this process.

VI. Grounds of Rejection to be Reviewed on Appeal

Claims 1-16, 18-26 and 29-32 stand rejected under 35 U.S.C. §102(b) as being anticipated by Netscape Application Builder, User Guide, 1999 ("NAB").

The rejection of independent claims 1, 10, and 21 as being anticipated by NAB under 35 U.S.C §102(b) are at issue in this appeal.

VII. Argument

The present invention provides an application platform having a set of features and services designed to facilitate the development and use of business applications useable on any platform that supports standard server-side presentation logic. The Appellant's invention claims an application platform that lies between commerce application software and the application's server to form a universal interface. This allows the application software and the server on which it runs to be platform independent. Thus the data repository housing the data elements, the applications software, and the presentation data are all platform independent.

To better appreciate the context of the present invention and its rejection, a brief history of the relationship between the assignees of the present invention, Sun Microsystems, Inc, ("Sun") and Netscape Communications Corp. ("Netscape") is provided. A friendly working relationship between Sun and Netscape has long existed. As early as 1995, Sun and Netscape began working together to foster networking solutions. Indeed, in April of 1997 Sun and Netscape jointly developed Java Foundation classes. In June of 1997 Netscape selected Sun's JavaBeans™ as Component Model for

Netscape ONE Platform. In 1999, Sun and Netscape entered into an alliance to foster development of an e-commerce platform. The present NAB was one of the original documents prepared by this alliance. Future renditions, including the present invention, would become known as iPlanet and serve to broaden the applicability of the platform by removing several of the earlier NAB limitations. Indeed the present invention was assigned to Sun by Netscape.

A. The rejection of claims 1, 10, and 21 Under 35 U.S.C. 102(b) are improper since NAB has not been shown to either explicitly or inherently disclose an applications platform that provides access to a data repository independent of the run-time platform.

The Examiner argues in the most recent rejection that a demonstration that a NAB application can run on a Netscape Application Server ("NAS") discloses the invention. Rather than identifying what portions of NAB discloses these limitations, the rejection simply concludes that anticipation is evident. Claim 1 (and claims 10 and 21 in varying language) states, among other things, "a software portion configured to provide access to, and caching of, data elements, including a data and object repository, independent of the run-time platform for providing commerce software applications; .... " (emphasis added).

The rejection of claim 1 argues that the NAB disclosure states, in a discussion about Enterprise JavaBeans ("EJBs"), "These applications can be written once and then deployed on any server platform that supports EJBs", and that this disclosure is anticipatory of access to the data elements independent of the run-time platform. (emphasis in original, see Final Rejection page 3) The Appellant disagrees. NAB does not provide for cross-platform support allowing software portability. NAB also does not possess the ability to plug into different backend data repositories using an abstract data layer with the application software running on top, i.e. independent of the run-time

platform for providing commerce software applications. This conclusion of anticipation is simply without merit, is conclusory, and thus wanting of a *prima facie* case of anticipation.

The rejection also argues that it is the Appellant's burden to show that NAB does not provide access to the data elements independent of the run-time platform or to present data by any server capable of server-side presentation logic. The Appellant disagrees. The burden only shifts upon the presentation of a *prima facie* case of anticipation. In this case such a showing is wanting. The Appellant contends that it remains the Examiner's burden to show that NAB discloses what is claimed, and only upon such a showing is it the Appellant's burden to refute such an allegation. The Examiner has not shown that the aforementioned limitations are disclosed in NAB. The Examiner, rather, concludes, with faulty logic, that such limitations are presumably inherent.

The Examiner attempts to argue that the Appellant's own use of the NAS is an example of a server supporting server-side presentation logic and that this example conveys the meaning that NAB is applicable to any platform. To be clear, NAB provides a means to build applications for NAS. NAS does support server-side presentation logic and EJB, but that fact does not mean that NAB is applicable to any server that supports server-side presentation logic. The only conclusion that can be properly drawn is that NAB can facilitate the creation of applications operative on NAS.

This is not what is claimed. The limitation as claimed is that the application platform provides access to a data repository independent of the run-time platform. NAB does not disclose such a capability nor does the Examiner make any effort to recite a portion of NAB that does so. The rejection is improper and should be withdrawn.

B. The rejection of claims 1 and 10 Under 35 U.S.C. 102(b) are improper since NAB fails to either explicitly or inherently disclose an applications platform that provides static and dynamic presentation data for presentation by any server capable of server-side presentation logic.

As mentioned, the claimed invention is a run-time platform operatively adaptable to any server platform capable of server-side presentation logic. The run-time platform claimed by the Appellant is far more flexible and useful than that described in the NAB. Equating applications that can be displayed on any server that supports EJBs, as done by the Examiner, to a run-time platform that is adaptable to any server platform capable of server-side presentation logic is incorrect and in violation of the anticipation provisions of MPEP 2131. The question becomes, are EJBs, as described in the NAB, adaptable to any server capable of server-side presentation logic. The answer to the question is resoundingly NO. As stated in NAB, the applications developed by NAB are adaptable to any server supporting EJBs.

To be anticipated, each element of a claim must be disclosed either expressly or inherently and no element of a claim can be ignored. The classic test for anticipation, as cited in *Lewmar Marine, Inc. v. Barient, Inc.*, 827 F.2d 744 (Fed. Cir. 1987) *cert. denied* 484 U.S. 1007 (1988), states "that which literally infringes if later, anticipates if earlier." NAB, if it were written after the present application, would not infringe the present application as currently claimed. The presentation logic is specific to a NAS and not any server that supports server-side presentation logic. While one may be tempted to argue that JAVA and EJBs are platform independent, it would be incorrect to apply that association with NAB, a software development tool that is specifically written for the Netscape environment. For a limitation to be inherent in a reference, it must naturally follow from what is disclosed. A limitation that may follow, is possible, or even obvious, is not inherent. Mere probability or possibility are irrelevant to the analysis of

anticipation. As the NAB would not infringe the present application as claimed, it cannot anticipate the application.

The Examiner refers to NAB as disclosing (anticipating) the present invention. While NAB is clearly related to the present invention, it does not anticipate each and every limitation of the claimed invention. Prior to the present invention, applications built using NAB, as disclosed in the NAB, were only deployable on the NAS. *See* NAB C2-5. Other application builder tools known to one skilled in the art shared this type of limitation, i.e. being restricted to a particular server environment. The present invention provides and claims an interface and functional support as an intermediate layer between the application software and the server to allow application software developed from various tools to be implemented on any server environment. Thus the presentation data can be presented on any server capable of server-side presentation logic. Furthermore, the data accessed by the application can be from a repository that is independent of the run-time platform on which the application is running. This cross-platform support of the present invention is novel with respect to NAB.

The present invention's claimed ability to adapt to any application server supporting server-side presentation logic allows the application platform to be incorporated with a wide variety of application servers. Once the application platform is situated, any application developed for use on the platform is capable of running in the computing environment of whichever application server has been used. NAB does not provide this versatility. The Office Action of October 23, 2006 argues that NAB discloses this functionality by its description of using EJBs. The Office Action states "The applications [EJB applications] can be written once and then deployed on any server platform that supports EJBs." (emphasis original). *See* NAB C9-1. Thereafter, the Office Action concludes that NAB applies to any platform. However, NAB clearly states that EJB applications can be written once and then displayed on any server platform that supports EJBs. Server platforms that support EJBs and those that support server-side presentation logic are not necessarily the same. Many servers exist that support server-

side presentation logic that do not support EJB. NAB describes a layer deployable on servers that only support EJBs, i.e. a NAS.

C. Conclusion

As the NAB was written and developed by the assignee of the present invention, it is reasonable to assume that the present invention and the NAB are related. However, NAB is but one application building tool that can be used to produce applications operable on one of the Appellant's platforms. While there are certainly similarities between the present invention and NAB, and applications built using NAB can operate on the present application platform, the present invention discloses and claims an application platform that provides a run-time environment compatible with a wide variety of computing environment and operating systems well beyond those disclosed by NAB.

Perhaps the best evidence of the limited applicability of NAB to NAS is Netscape's own news release with respect to NAB. Netscape stated in its September, 1999 release of NAB,

> Powerful New Netscape Application Builder 3.0 Simplifies Development of Multi-Tier, Distributed, Business-Critical Applications for Deployment on Netscape Application Server
>
> MOUNTAIN VIEW, Calif., Sept. 22 /PRNewswire/ -- Netscape Communications Corporation (Nasdaq: NSCP) today announced Netscape Application Builder 3.0, a powerful new web development environment for building multi-tier distributed applications deployed on Netscape Application Server software. With Netscape Application Builder 3.0, developers can quickly and easily build different tiers of an application that leverage the rich application and infrastructure services delivered by Netscape Application Server. ...

In view of all of the above, claims 1 and 10 are believed to be allowable and the case in condition for allowance. Appellant respectfully requests that the Examiner's rejections based on 35 U.S.C. §102(b) be reversed for the pending claims.

Respectfully submitted,

Date: 1/14/08

Michael C. Martensen, No. 46,901
Hogan & Hartson LLP
One Tabor Center
1200 17th Street, Suite 1500
Denver, Colorado 80202
(719) 448-5910 Tel
(303) 899-7333 Fax

VIII. CLAIMS APPENDIX

1. A computer system run-time platform for providing features and services for commerce software applications, and operatively adaptable to any server platform capable of server-side presentation logic, the applications platform comprising:

a software portion configured to provide access to, and caching of, data elements, including a data and object repository, independent of the run-time platform for providing commerce software applications;

a software portion configured to inherit hierarchical application logic from the commerce applications platform;

a software portion configured to provide static and dynamic presentation data for presentation by any server capable of server-side presentation logic;

a software portion configured to maintain permanent and session application data persistent across user request boundaries during a single user session; and

a software portion configured to enable access to a business object during the user session.

2. The computer system platform of claim 1, wherein the data elements are stored within a computer-readable medium in the form of a data structure forming a list of at least one data element, wherein each data element comprises:

a first field containing data representing a data element name;

a second field containing data representing the data element type; and

a third field containing data representing the data element value.

3. The computer system platform of claim 1, further comprising a software portion configured as a rule engine for evaluating rule parameters.

4. The computer system platform of claim 1, further comprising a data management software portion configured to store and retrieve data during a user session.

5. The computer system platform of claim 1, further comprising user a software portion configured to provide or deny a user access to the commerce software applications.

6. The computer system platform of claim 1, further comprising a software portion configured to transfer data to and from a data store.

7. The computer system platform of claim 6, wherein the data store further comprises LDAP data stores.

8. The computer system platform of claim 6, wherein the data store further comprises database data stores.

9. The computer system platform of claim 1, wherein the business object is cached during the user session.

10. A method for implementing a first software application resident on a commerce application platform wherein the commerce application's platform is a run-time platform configured to provide access to data elements, hierarchical inheritance of the first software application logic, static and dynamic presentation data, access to business objects, and access to permanent and session application data during a single user session, the method comprising:

providing presentation information by the first software application seeking input data from a user;

receiving input data from a data repository for use by the first software application, wherein the data repository is independent of the commerce application platform;

passing the input data to the commerce application platform for validation;

validating the data by the commerce application platform;

providing, by the commerce application platform, business object functionality to the application;

preparing presentation information by the application based upon the business object functionality for presentation by any server capable of server-side presentation logic; and

accessing the permanent and session application data persistent across user request boundaries during the single user session.

11. The method of claim 10, wherein the step of providing presentation information further comprises providing static and dynamic presentation data.

12. The method claim 10, wherein the passing of input data further comprises passing user identification information.

13. The method of claim 10, wherein the passing of input data further comprises passing data corresponding to commerce functionality.

14. The method of claim 10, wherein the step of validating the data further comprises invoking a rule engine to determine a validation result.

15. The method of claim 10, further comprising the step of creating, by the commerce application platform, a business object for providing business functionality.

16. The method of claim 10, further comprising the step of accessing, by the commerce application platform, an existing business object.

18. The method of claim 10, further comprising the step of implementing a second software application on the commerce application platform.

19. The method of claim 18, further comprising the step of implementing a second software application by concurrently implementing the first software application and the second software application.

20. The method of claim 18, further comprising the step of accessing a business object by both the first and the second software applications.

21. A method for providing services to a first software application residing on a commerce application platform wherein the commerce applications platform is a run-time platform configured to provide access to data elements, hierarchical inheritance of the first software application logic, static and dynamic presentation data for presentation by any server capable of server-side presentation logic, access to business objects, and access permanent and session application data during a single user session, the method comprising:

receiving from the application input data for validation from a data repository, wherein the data repository is independent of the commerce application platform;

validating the input data;

providing business object functionality to the application; and

accessing the permanent and session application data persistent across user request boundaries during the single user session.

22. The method of claim 21, wherein the input data received from the application relates to a commerce application function.

23. The method of claim 21, wherein the input data received from the application includes user identification information.

24. The method of claim 21, wherein the step of validating the input data further comprises invoking a rule engine to determine a validation result.

25. The method of claim 21, further comprising the step of creating a new business object.

26. The method of claim 21, further comprising the step of accessing an existing business object.

29. The method of claim 21, further comprising the step of creating a persistent object based on a persistent object framework.

30. The method of claim 21, further comprising the step of receiving input data from a second application on the commerce application platform.

31. The method of claim 21, wherein the step of receiving input data from the first software application further comprises concurrently receiving input data from a second software application.

32. The method of claim 21, wherein the step of providing business object functionality to the application further comprises providing the same business object functionality to a second software application.

IX. EVIDENCE APPENDIX

No copies of evidence are required with this Appeal Brief. Appellant has not relied upon any evidence submitted under 37 C.F.R. §§ 1.130, 1.131, or 1.132.

X. RELATED PROCEEDINGS APPENDIX

There are no copies of decisions rendered by a court or the Board to provide with this Appeal as there are no related proceedings.